Pimp my ASUS WL500GP V2 using OpenWrt

Jan Helber

This documentation describes howto setup an OpenWrt build environment, compile a customized OpenWrt image and flash it on the router using the internal boot loader.

Powered by LaTeX
Generated on July 17, 2010 at 20:26

Contents

1 Introduction to OpenWrt **1**

2 Setup build environment **1**

2.1 Setup package repository . . . 3

2.2 Adding packages . . . 3

2.2.1 Loop AES . . . 3

2.2.2 Madplay . . . 5

2.2.3 Samba . . . 6

2.2.4 VPN . . . 6

2.2.5 MISC . . . 7

3 Compile OpenWrt **8**

4 Download OpenWrt binary **8**

5 Flash router with OpenWrt **8**

6 Minimal system setup **9**

6.1 Activate SSH . . . 10

6.2 Wireless LAN . . . 11

6.3 Wireless client mode . . . 12

7 Network configuration **15**
7.1 Normal routing functionality . . . 15
7.2 Two separate subnets . . . 16
7.3 Static IP (MAC based) . . . 17
7.4 Firewall . . . 18
7.5 Load the new network configuration . . . 19

8 Setup your system **19**
8.1 opkg . . . 19
8.2 USB 1.1 & USB 2.0 support for mass storage devices . . . 20
8.3 Partition and format USB-storage . . . 21
8.4 Mount USB-storage . . . 24
8.5 Activate swap partition . . . 26

9 Packete installieren **26**
9.1 Installation von Packeten auf USB-storage vorbereiten . . . 26
9.2 Loop AES . . . 27
9.3 Samba . . . 29
9.4 Cron . . . 31
9.4.1 DynDns . . . 33
9.4.2 Rsync . . . 33
9.4.3 Zeitzone . . . 34
9.4.4 RDate . . . 34
9.4.5 OpenVPN . . . 34
9.4.6 Socks5 . . . 37
9.4.7 WOL . . . 37
9.4.8 QoS . . . 37
9.5 Shell-Skripte . . . 39
9.6 MAC address cloning . . . 42
9.7 SSH-Public-Key-Authentification . . . 42

10 Backup **43**

11 Viel Spaß! **43**

12 FAQ **44**

13 Anhang 44

14 Quellen 44

15 Marken 45

16 Haftungsausschluss 45

List of Figures

List of Tables

1 Introduction to OpenWrt

Why would you flash a customized operating system to a router by risking to break it? I don't know - there are probably lot's of reasons - my use case was the following:

I wanted to have a decentral RAID 1 with 500 GB of mirrored disk space in two households. Therefore I extended two routers with 1TB of storage for each and setup cronjobs for incremental backups: Each night half of the disk space is mirrored from Router 1 to Router 2 and the the other half is mirrored the other way round.

2 Setup build environment

If you don't want to compile OpenWrt by your own, just skip this step and continue at section 4 on page 8.

First of all you need to download the latest version of OpenWrt via SVN to your Linux PC. In my case it was revision 21773 and I downloaded it to ~/openwrt-ASUSv2-latest [ASUS WL-500gp V2]:

```
username="`(whoami) 2>/dev/null | awk '{ printf("%s", $1); }'`";
if [ "root" == $username ]; then
clear
  for ((  i = 0 ;  i <= 5;  i++  ))
  do
    echo For compiling the sources please change to a non-admin user! ;
  done
else
  mkdir ~/openwrt-ASUSv2-21773
  cd ~/openwrt-ASUSv2-21773
  svn co svn://svn.openwrt.org/openwrt/trunk -r 21773 kamikaze;
  svn co svn://svn.openwrt.org/openwrt/packages -r 21773 packages;
fi
```

Next the prerequisites need to be checked. The following command only informs about missing programs, but they need to be installed manually:

```
make prereq
```

On my setup not all necessary programs were available:

```
sudo apt-get install gcc original-awk flex patch autoconf \
bison libncurses5-dev zlib1g-dev gawk g++ sdcc gettext ruby \
fastjar jikes libssl-dev atftp texinfo
```

With the following command you may configure your build environment before compiling the binary which will be flashed to the router (marked with *) and also the packages available as module for later installation on the router (marked with M):

```
cd ~/openwrt-ASUSv2-21773/kamikaze
make menuconfig
```

I configured my setup as follows:

- OpenWrt Configuration
  - Target System (Broadcom BCM947xx/953xx [2.4])
  - Target Profile (ASUS WL-500g Premium)
- Global build settings
  - Select all packages by default
- Kernel modules > Filesystems
  - kmod-fs-ext2 <*>
  - kmod-fs-ext3 <*>
  - kmod-fs-vfat <*>
- Kernel modules > USB Support
  - kmod-usb-core <*>
  - kmod-usb-ohci <*>
  - kmod-usb-storage <*>
  - kmod-usb-uhci-iv <*>
  - kmod-usb2 <*>

Afterwards exit and save your configuration.

Depending on the packages that you selected for your OpenWrt build some more prerequisites might be necessary. Therefore the prerequisites should be rechecked in advance to the 1st build:

```
cd ~/openwrt-ASUSv2-21773/kamikaze
make prereq
```

2.1 Setup package repository

For being able to install packages on the router from the linux workstation a repository needs to be setup. As long as there is already a running apache server this can be done quite easily:

```
# By adding a symlinc
ln -s ~/openwrt-ASUSv2-21773/kamikaze/bin/brcm-2.4 \
/var/www/trunk-21773
# Don't forget to check apache configuration...
# Options FollowSymLinks +Indexes

chmod o+rx -R ~/openwrt-ASUSv2-21773/kamikaze/bin/brcm-2.4

# Or by adding an alias to the apache configuration
# Alias /trunk-21773/ \
# "/home/user/openwrt-ASUSv2-21773/kamikaze/bin/brcm-2.4/"
```

2.2 Adding packages

```
cd ~/openwrt-ASUSv2-21773/kamikaze

make download;
make V=99 world &> ../make.log;

clear
ls -la ~/openwrt-ASUSv2-21773/make.log
ls -la ~/openwrt-ASUSv2-21773/kamikaze/bin/\
brcm-2.4/openwrt-brcm-2.4-squashfs.trx
```

2.2.1 Loop AES

Before we install the loop-aes, some comments on building OpenWrt from SVN. The latest official release of OpenWrt - in my case KAMIKAZE (bleeding edge, r21771) - doesn't include loop-aes. Fortunately, the current SVN version DOES include a patched loop kernel module ("kmod-loop-aes"), but no patched userland tools. So, with the SVN version we have to go only half the way to an encrypted system by patching the userland tools. [Filesystem Encryption]

First check in ~/openwrt-ASUSv2-21773/kamikaze/package/util-linux-ng/Makefile which version of util-linux-ng is included in the svn (mine was PKG_VERSION:=2.13.0.1). Then download the corresponding loop-aes version (in my case: loop-AES-v3.2b):

```
cd ~/openwrt-ASUSv2-21773
wget http://loop-aes.sourceforge.net/loop-AES/loop-AES-v3.2b.tar.bz2
tar -jxvf loop-AES-v3.2b.tar.bz2
```

There's a file included: util-linux-ng-XXX.diff (in my case util-linux-ng-2.13.0.1.diff). Copy it to kamikaze/package/util-linux-ng/patches and rename it to something like xxx-util-linux-ng-2.13.0.1.diff depending on how many patches there are already (in my case there was already:

```
cd ~/openwrt-ASUSv2-21773/loop-AES-v3.2b

ls ~/openwrt-ASUSv2-21773/kamikaze/package/util-linux-ng/patches
# In my case there was only one patch (001-cris_avr32_label.patch)

cp util-linux-ng-2.13.0.1.diff ~/openwrt-ASUSv2-21773/kamikaze/\
package/util-linux-ng/patches/002-util-linux-ng-2.13.0.1.diff
```

Next, edit kamikaze/package/util-linux-ng/Makefile and replace all occurrences of "losetup", "mount-utils" and "swap-utils" with "aeslosetup", "aesmount-utils", "aesswap-utils" (not in Build/Compile section of course). We want to avoid conflicts with the original packages and distinguish them better from the original packages. (In fact, it is possible to clone the whole package and build a patched and unpatched version simultaneously.):

```
vi ~/openwrt-ASUSv2-21773/kamikaze/package/util-linux-ng/Makefile

#:1,122s/mount-utils/aesmount-utils/gi
#:1,122s/losetup/aeslosetup/gi
#:1,122s/swap-utils/aesswap-utils/gi
#:129,$s/mount-utils/aesmount-utils/gi
#:129,$s/losetup/aeslosetup/gi
#:129,$s/swap-utils/aesswap-utils/gi
```

Since aeslosetup doesn't exist till now we simply create a symlink.
Otherwise the build-process would return with an error:

```
cd vi ~/openwrt-ASUSv2-21773/kamikaze

ln -s ~/openwrt-ASUSv2-21773/kamikaze/build_dir/\
target-mipsel_uClibc-0.9.30.1/util-linux-ng-2.13.0.1/mount/losetup \
~/openwrt-ASUSv2-21773/kamikaze/build_dir/\
target-mipsel_uClibc-0.9.30.1/util-linux-ng-2.13.0.1/mount/aeslosetup

ls -la ~/openwrt-ASUSv2-21773/kamikaze/build_dir/\
target-mipsel_uClibc-0.9.30.1/util-linux-ng-2.13.0.1/mount/ \
| grep aeslosetup
```

It might happen that you get an error message that aeslosetup would not exist after the first compile. Simply recreate the symlink and compile again. After the second build run the symlink should stay and the issue should not appear again.

Before loop-aes can be selected in menuconfig it needs to be linked to the packages directory below kamikaze folder:

```
cd ~/openwrt-ASUSv2-21773/kamikaze/package
ln -s ../../packages/utils/loop-aes .

./scripts/feeds update
./scripts/feeds install loop-aes
```

Select all necessary packages:

```
make menuconfig
```

I configured my setup as follows:

- Kernel modules > Block Devices
  - kmod-loop <N> excludes
- Kernel modules > Cryptographic API modules
  - kmod-crypto-core <*>
  - kmod-crypto-aes <*>
  - kmod-loop-aes <*>
- Utilities
  - aeslosetup <Y> includes
  - aesmount-utils <Y> includes
- Utilities > disc
  - aesswap-utils <M> builds as package

Exit and save your configuration.

Due to dependencies it might happen that kmod-loop-aes can only be compiled as module on the first run and not into the OpenWrt binary. As soon as aeslosetup was selected to be compiled into the OpenWrt binary kmod-loop-aes should work as well.

2.2.2 Madplay

First we need to figure out the dependencies of this package using the following command:

```
cd ~/openwrt-ASUSv2-21773
grep DEPENDS packages/sound/madplay/Makefile
# DEPENDS:=+libid3tag +libmad

grep DEPENDS packages/libs/libmad/Makefile packages/libs/libid3tag/Makefile
# DEPENDS:=+zlib
```

zlib is provided by the kamikaze distribution, in kamikaze/package/zlib, so we don't have to mess with that.

Provide symlinks in kamikaze/package, to the packages you want to build:

```
cd kamikaze/package
ln -s ../../packages/sound/madplay .
ln -s ../../packages/libs/libmad .
ln -s ../../packages/libs/libid3tag .
```

2.2.3 Samba

```
cd ~/openwrt-ASUSv2-21773/kamikaze/package
ln -s ../../packages/network/samba .

cd ~/openwrt-ASUSv2-21773/kamikaze
./scripts/feeds update
./scripts/feeds install samba3
./scripts/feeds install samba-server
```

Select all necessary packages:

```
make menuconfig
```

I configured my setup as follows:

- Network
  - samba3 <M> package

Exit and save your configuration.

2.2.4 VPN

[OpenWrt OpenVPN] [DDWrt OpenVPN]

```
cd ~/openwrt-ASUSv2-21773/kamikaze
./scripts/feeds update
./scripts/feeds install openvpn
./scripts/feeds install openswan
#./scripts/feeds install strongswan
#./scripts/feeds install strongswan4
```

Select all necessary packages:

```
make menuconfig
```

I configured my setup as follows:

- Network > VPN
  - everything <M> package

Exit and save your configuration.

2.2.5 MISC

```
cd ~/openwrt-ASUSv2-21773/kamikaze/package
ln -s ../../packages/mail/mutt .
ln -s ../../packages/utils/zip .
ln -s ../../packages/utils/vim .
ln -s ../../packages/utils/unzip .
ln -s ../../packages/utils/unrar .
ln -s ../../packages/utils/tar .
ln -s ../../packages/utils/gzip .
ln -s ../../packages/utils/bzip2 .
ln -s ../../packages/utils/sed .
ln -s ../../packages/utils/at .
ln -s ../../packages/net/openssh .

cd ~/openwrt-ASUSv2-21773/kamikaze
./scripts/feeds update
./scripts/feeds install rsync
./scripts/feeds install etherwake
./scripts/feeds install tcpdump
./scripts/feeds install ssldump
./scripts/feeds install tcpdump-mini
./scripts/feeds install usbutils
./scripts/feeds install lsof
./scripts/feeds install sed
./scripts/feeds install unfs3
./scripts/feeds install vim-full
./scripts/feeds install aircrack-ng
./scripts/feeds install aircrack-ptw

./scripts/feeds list | more
```

Select all necessary packages:

```
make menuconfig
```

I configured my setup as follows:

- Network
  - rsync <M> package
- Libraries
  - libpopt <M> package

Exit and save your configuration.

3 Compile OpenWrt

```
cd ~/openwrt-ASUSv2-21773/kamikaze;

make prereq;
make download;

date > ../make.log; make V=99 world >> ../make.log 2>&1; date >> ../make.log;

clear
ls -la ~/openwrt-ASUSv2-21773/make.log;
ls -la ~/openwrt-ASUSv2-21773/kamikaze/bin/\
brcm-2.4/openwrt-brcm-2.4-squashfs.trx;
```

Since you compiled OpenWrt by your own, you may skip the step downloading OpenWrt binary and continue at section 5 on page 8.

4 Download OpenWrt binary

http://www.helber.info/fileadmin/Projekte/Router/trunk-21773/openwrt-brcm-2.4-squashfs.trx
http://www.helber.info/fileadmin/Projekte/Router/trunk-21773/packages

5 Flash router with OpenWrt

For flashing the recently compiled (or downloaded) openwrt-brcm-2.4-squashfs.trx binary the network of the workstation need to be setup with a static IP (192.168.1.100). Then the workstation shall be connected via LAN cable (CAT5) with port 1 of the router. The following should do the trick (diagnostic mode):

- Unplug power cable of router
- Press and keep pressing Restore push button

- Plug in the power cable
- Wait till power light flashes at 1Hz
- Release the Restore push button
- Execute the tftp command on workstation

Linux:

```
# Set static IP in Debian/Ubuntu/Kubuntu
/etc/init.d/network-manager stop;
ifconfig eth0 up;
ifconfig eth0 192.168.1.100

# Flash the image
atftp --trace --option "timeout 1" --option "mode octet" --put --local-file \
~/openwrt-ASUSv2-11765/kamikaze/bin/openwrt-brcm47xx-squashfs.trx 192.168.1.1
```

Windows/Dos:

```
tftp -i 192.168.1.1 PUT D:\Projekte\2010-06-08-OpenWrt-ASUSv2\openwrt-brcm-2.4-squash
# Übertragung erfolgreich: 2428928 Bytes in 14 Sekundens, 173494 Bytes/s
```

Now you should wait a while. If you are connected via serial port you will see when flashing the image is finished. If you don't have a serial connection you should be safe by waiting 6 minutes (wiki suggests this). Afterwards restart the router (power off & power on).

6 Minimal system setup

After flashing and restarting the router you need to log in via telnet - no password is needed for this:

```
telnet 192.168.1.1
```

For questions in bulletin boards regarding issues with your router it is often necessary to know which version you are using. You can figure this out with the following command:

```
uname -a

# In my case it was:
# Linux OpenWrt 2.4.37.9 #18 Tue Jun 15 00:52:02 CEST 2010 mips GNU/Linux
```

6.1 Activate SSH

The first thing that should be done is to set a new password with the command "passwd". This will automatically deactivate the telnet daemon and activate the SSH damon (dropbear):

```
passwd
```

Afterwards give it a minute to generate the keys and setup dropbear (done automatically). Then after closing the telnet connection with CTRL+D it should be possible to establish a connection using ssh:

```
ssh 192.168.1.1
```

This is for my personal use. *DO NOT COPY THIS* since this is my public key and you would make your router accessible to me:

```
echo "ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAgBVMmdbUcuCgJo2fMO9UEGmscDS8E2do1T\
Ig1n6B4IDGTdro63IGXnxnIKZ4weCiuhGXQcXKFhAXqsUhG0lD2++9oHWvqYzOwZdFWsaFyV5\
TjmqI3bqbnB7KqI6ZV/SwJpYH3g/57BBC5KUaN1L1dvQ42nHtNbiJUNAFK9qLPWmc/3M1bAre\
+CPpgneYfzM/KjY0kluHdBpQOm/KxmuaSPC5XRGGlyKG1HL9RfvgjNoPnKEeiFgMc+ZjhLMtS\
+M9xku592Uw+w9STB6lsPrIbViLo0ODLIEBrgMGEIcy8kxZkTQ7XEgw2yJfU1UmAlIoIrxuaq\
tuPB8eBNhCBF+y7WJgjUB0UsK29fljSsnCbeS6AktbLd22psHbZ5Szq11jl/RpCpuXAGp5QlK\
suO8gYFWi7oJI8jcmxyasOmUAy1LSDqEkYtGVX59LVvKHhAv8TZJb6UYL6NkPWbHAKRs7wJil\
gtQ48JT8YypPLTZchEUKREV+H7BSW568cCV+zeHu5RHeJjU18GX7tLksj/Tf5PGlkeJIu+cRA\
3guU6b6PhsttxbXuptOVq+RQWziOgIlQf5Hk49kZEACptqzT09+Fq8/5kMDjTlMaj20B8rm9o\
H5eg81b3g4hq8bQa4aW3N4hs59NRv2zvWqREp7wfr9xR3inhl4mb+N43JxJPlqYcNDQ== rsa\
-key-20090823" > /etc/dropbear/authorized_keys;
/etc/init.d/dropbear restart;
```

6.2 Wireless LAN

To set up the WLAN with WPA[1] encryption, the file /etc/config/wireless needs to be edited. An example how this is to be done is listed below. Of course you have to replace "12345" with your own password.

```
echo "config wifi-device  wl0
        option type     broadcom
        option channel  5

        # REMOVE THIS LINE TO ENABLE WIFI:
        option disabled 1

config wifi-iface
        option device     wl0
        option network    lan
        option mode       ap
        option ssid       OpenWrt
        option encryption psk
        option key        12345" > /etc/config/wireless

# Load the new settings
wifi down
wifi up
iwconfig
```

[1] wpa2 might cause trouble with XP SP2 and cheap WLAN devices

6.3 Wireless client mode

[Client Mode Wireless] [Bridged Client]

Before doing any actual configuration, wpa-supplicant needs to be installed and the wifi interface must be enabled in order to be able to scan for networks in the vincinity:

```
# Install wpa-supplicant on flash
# (should work without hard disk as well)
opkg install --dest root wpa-supplicant

# Remove the disable 1 option from the wireless configuration
uci del wireless.wl0.disabled

# Save changed configuration file
uci commit wireless

# Start wireless using the wifi command
wifi
```

Now we can issue the iwlist scan command to list networks in range:

```
iwlist wl0 scan
```

Edit /etc/config/wireless and change the mode of the existing wireless network to sta, the channel and the ESSID to what you observer from iwlist call. In my case:

```
echo "config 'wifi-device' 'wl0'
        option 'type' 'broadcom'
        option 'channel' '9'

        # REMOVE THIS LINE TO ENABLE WIFI:
        option disabled 0

config 'wifi-iface'
        option 'device' 'wl0'
        option 'network' 'lan'
        option 'mode' 'sta'
        option 'ssid' 'OpenWrt'
        option 'encryption' 'psk'
        option 'key' 'o34Fgc78Jxa3HjkMn'" > /etc/config/wireless
```

Proceed with calling wifi to apply the new wireless configuration and check the result using iwconfig:

```
wifi
iwconfig
```

```
#wl0       IEEE 802.11-DS  ESSID:"OpenWrt"
#          Mode:Managed  Frequency:2.447 GHz  Access Point: 00:22:15:32:9E:66
#          Bit Rate=36 Mb/s   Tx-Power:32 dBm
#          RTS thr:off   Fragment thr:off
#          Power Management:off
#          Link Quality=5/5  Signal level=-11 dBm  Noise level=-92 dBm
#          Rx invalid nwid:0  Rx invalid crypt:9645  Rx invalid frag:0
#          Tx excessive retries:4  Invalid misc:0   Missed beacon:0
```

Access Point: 00:22:15:32:9E:66 indicates a successful connection.

At this point hosts connected to the LAN ports of the OpenWrt router should be able to receive DHCP directly from the remote access point.

Since the device is operating as client in another network and relays all communication from the associated Access Point to its LAN hosts, the local DHCP server should be disabled to avoid collisions.

Edit /etc/config/dhcp and set the predefined DHCP pool for the 4 LAN ports to ignore. Only on the WAN port a DHCP daemon shall offer IPs for us always being able to connect to the router:

```
echo "config dnsmasq
        option domainneeded     1
        option boguspriv        1
        option filterwin2k      '0'  #enable for dial on demand
        option localise_queries 1
        option local    '/wan/'
        option domain   'wan'
        option expandhosts      1
        option nonegcache       0
        option authoritative    1
        option readethers       1
        option leasefile        '/tmp/dhcp.leases'
        option resolvfile       '/tmp/resolv.conf.auto'
        #list server            '/mycompany.local/1.2.3.4'
        #option nonwildcard     1
        #list interface         br-lan
        #list notinterface      lo

config dhcp lan
        option interface        lan
        option start    100
        option limit    150
        option leasetime        12h
        option ignore   1

config dhcp wan
```

```
        option interface        wan
        option start    100
        option limit    150
        #option ignore   1
        option leasetime        12h" > /etc/config/dhcp

# Apply the change by restarting dnsmasq:
/etc/init.d/dnsmasq restart
```

Setup 2 VLANs. One for the 4 LAN ports in bridged mode where the IP shall be acquired via DHCP. The other VLAN is for the WAN port which will offer IPs using a DHCP daemon for us always being able to connect to the router (diagnostic purpose):

```
echo '#### VLAN configuration
config switch eth0
        option enable   1

config switch_vlan eth0_0
        option device   "eth0"
        option vlan     0
        option ports    "0 1 2 3 5"

config switch_vlan eth0_1
        option device   "eth0"
        option vlan     1
        option ports    "4 5"

#### Loopback configuration
config interface loopback
        option ifname   "lo"
        option proto    static
        option ipaddr   127.0.0.1
        option netmask  255.0.0.0

#### LAN configuration
config interface lan
        option type     bridge
        option ifname   "eth0.0"
        option proto    dhcp

#### WAN configuration
config interface        wan
        option type     bridge
        option ifname   "eth0.1"
        option proto    static
        option ipaddr   192.168.4.1
```

```
        option netmask  255.255.255.0' > /etc/config/network

/etc/init.d/network restart
```

This step is not strictly required but disabling the firewall saves resources when operating as a dumb bridge - there is no need to filter traffic running from wifi to ethernet and back.

Stop and disable the firewall by using the init script:

```
/etc/init.d/firewall stop
/etc/init.d/firewall disable
```

7 Network configuration

7.1 Normal routing functionality

For setting up normal routing functionality with 1 external WAN port (DHCP client) and 4 internal LAN ports (DHCP daemon 192.168.4.x) the following setting may be used:

```
echo '#### VLAN configuration
config switch eth0
        option enable   1

config switch_vlan eth0_0
        option device   "eth0"
        option vlan     0
        option ports    "0 1 2 3 5"

config switch_vlan eth0_1
        option device   "eth0"
        option vlan     1
        option ports    "4 5"

#### Loopback configuration
config interface loopback
        option ifname   "lo"
        option proto    static
        option ipaddr   127.0.0.1
        option netmask  255.0.0.0

#### LAN configuration
config interface lan
        option type     bridge
        option ifname   "eth0.0"
        option proto    static
```

```
        option ipaddr   192.168.4.1
        option netmask  255.255.255.0


#### WAN configuration
config interface        wan
        option ifname   "eth0.1"
        option proto    dhcp' > /etc/config/network

/etc/init.d/network restart
```

7.2 Two separate subnets

e.g. router direct connected with Internet:

```
echo "#### VLAN configuration
config switch eth0
        option vlan0    \"1 2 3 4 5*\"
        option vlan1    \"0 5\"


#### Loopback configuration
config interface loopback
        option ifname   \"lo\"
        option proto    static
        option ipaddr   127.0.0.1
        option netmask  255.0.0.0


#### LAN configuration
config interface lan
        option type     bridge
        option ifname   \"eth0.0\"
        option proto    static
        option ipaddr   192.168.1.1
        option netmask  255.255.255.0

#### WAN configuration
config interface        wan
        option ifname   \"eth0.1\"
        option proto    dhcp" > /etc/config/network
```

Switched network (behind another router):

```
echo "#### VLAN configuration
config switch eth0
```

```
        option vlan0    \"1 2 3 4 5\"
        option vlan1    \"0 5\"


#### Loopback configuration
config interface loopback
        option ifname   \"lo\"
        option proto    static
        option ipaddr   127.0.0.1
        option netmask  255.0.0.0


#### LAN configuration
config interface lan
        option type     bridge
        option ifname   'eth0.0'
        option proto    static
        option ipaddr   192.168.1.112
        option netmask  255.255.255.0
        option 'gateway' '192.168.1.1'
        option 'dns'    '192.168.1.1'

#### WAN configuration
#config interface        wan
#        option ifname   \"eth0.1\"
#        option proto    dhcp" > /etc/config/network
```

7.3 Static IP (MAC based)

http://johnbokma.com/mexit/2008/09/03/dhcp-static-ip-dnsmasq.html To make it easier to reach each device in the network, each device should have its own IP-address. An example how this is to be done is listed below:

```
echo "# Siemens
00:90:96:00:00:00 192.168.4.100
# Laptop1 WLAN
00:18:de:e2:08:61 192.168.4.201
# Laptop1 LAN
00:15:c5:c1:18:b8 192.168.4.202
# Laptop2
00:13:49:70:54:81 192.168.4.203
# Serverhome
00:30:05:8a:51:ff 192.168.4.240" > /etc/ethers
```

You should use the MAC adresse of your server were Openwrt-Apache-repository is running on instead of 00:30:05:8a:51:ff.

To make it even more easy to reach each device in the network, each device should have an easy to remember hostname:

```
echo "127.0.0.1 localhost. OpenWrt
192.168.4.1   asus
192.168.4.100 siemens
192.168.4.201 laptop1wlan
192.168.4.202 laptop1lan
192.168.4.203 laptop2
192.168.4.240 serverhome" > /etc/hosts
```

7.4 Firewall

If you want to allow SSH connections not only from inside the LAN - but also the WAN - this is to be done as follows [Simple Firewall] [Port Weiterleitung]:

```
echo "
# Accept SSH-port from wan
config rule
        option src              wan
        option dest_port        22
        option target           ACCEPT
        option proto            tcp" >> /etc/firewall.user
```

The above modified file /etc/firewall.user is being loaded from /etc/config/firewall.

If you want to tweak your firewall settings a bit you can do this by adding the following configuration.
The advantage of the settings below is that it is now also possible to establish a SSH connection to the router on port 443 besides port 22. In some networks the port 22 is blocked for outgoing connections due to security reasons. This is usually not the case for port 443 since it is reserved for HTTPS connections:

```
echo "
# Accept HTTPS-port from wan
config rule
        option src              wan
        option dest_port        443
        option target           ACCEPT
        option proto            tcp

# Redirect HTTPS-port requests from WAN to SSH-port
config redirect
       option src               wan
       option src_dport         443
       option dest              lan
```

```
        option dest_ip          192.168.4.1
        option dest_port        22
        option proto            tcp" >> /etc/firewall.user

/etc/init.d/firewall stop
/etc/init.d/firewall start
```

7.5 Load the new network configuration

For loading the new network configuration and bringing the WiFi up the following commands should help:

```
wifi up;
/etc/init.d/network restart;
```

Probably you will afterwards loose the network connection to the router. Simply setup the network interface of all workstations connected to the router to acquire the IP address using DHCP. Sometimes it is necessary to restart the router as well.

8 Setup your system

8.1 opkg

If you compiled all necessary packages by your own (section 3 on page 8), you should enter the following settings in file /etc/opkg.conf (eventually you need to adopt the IP address):

```
echo "src/gz snapshot http://192.168.4.240/trunk-21773/packages
#src/gz snapshots http://downloads.openwrt.org/snapshots/trunk/\
brcm-2.4/packages
dest root /
dest usb /mnt/part1/jffs_export
dest ram /tmp
lists_dir ext /var/opkg-lists
option overlay_root /overlay" > /etc/opkg.conf
```

The destination (dest roo) which is listed at first position will be taken as default (if destination is not set in opkg install call).

According to the configuration in section 7.3 *Static IP (MAC based)* on page 17 and section 2.1 *Setup package repository* on page 3 the IP of your Apache-OpenWrt-repository should be 192.168.4.240 and the repository should be located in the subfolder trunk-21773.

If you did not compile the source by your own but downloaded my OpenWrt binary in section 4 *download OpenWrt binary* on page 8, you can also use my package repository. In that case you should enter the following settings in file /etc/opkg.conf:

```
echo "src/gz snapshot http://www.helber.it/fileadmin/Projekte/Router/\
trunk-21773/packages
dest usb /mnt/part1/jffs_export
#src/gz snapshots http://downloads.openwrt.org/snapshots/trunk/\
brcm-2.4/packages
dest root /
dest ram /tmp
lists_dir ext /var/opkg-lists
option overlay_root /overlay" > /etc/opkg.conf
```

In case a package which you need is missing in my repository, please send me a short note (openwrt@helber.info). I will try to compile the package timely.

8.2 USB 1.1 & USB 2.0 support for mass storage devices

If you setup your environment like described in section 2 Setup build environment on page 1 then USB should work out of the box, since everything necessary is included in the OpenWrt binary.

There is lots what you can do with an USB connector on a router. e.g. USB-to-RS232, USB webcam, USB ethernet, USB bluetooth, USB VGA, USB sound ...
Have a look at: http://wiki.openwrt.org/OpenWrtDocs/Customizing/Hardware/USB

If you compiled the packages as modules, you can enable USB storages support with the following commands on the console [USB storage howto]:

```
# Without this command you might use old package-versions
opkg update
# USB common
opkg install kmod-usb-core
opkg install kmod-usb-ohci
# USB storage
opkg install kmod-scsi-core
opkg install kmod-usb-storage
# EXT3 and FAT32
opkg install kmod-fs-ext3
opkg install kmod-fs-vfat
# To enable lsusb
#opkg install usbutils
reboot
```

With the following commands on the console of the Asus router the necessary packages for USB 2.0 are being installed:

```
# USB 2.0 for my ASUS
opkg install kmod-usb2
opkg install kmod-nls-utf8 kmod-nls-iso8859-1
reboot
```

8.3 Partition and format USB-storage

Precondition: Section 8.2 on page 20.
To be able to partition the storage-device, fdisk needs to be installed:

```
opkg install fdisk
```

The idea of the following partitioning structure is the following:

- Part 1:
  - Filesystem: Ext3
  - Size: ca 91MB
  - Use: More space to install programs (internal mem is to less)
- Part 2:
  - Filesystem: Swap memory
  - Size: ca 200MB
  - Use: Especially for rsync of big partitions (on my setup 2x 460GB)
- Part 3:
  - Filesystem: Ext2 (AES encrypted)[2]
  - Size: $\frac{1}{2}$ of rest (on my setup 460GB)
- Part 4:
  - Filesystem: Ext2 (AES encrypted)
  - Size: $\frac{1}{2}$ of rest (on my setup 460GB)

If an USB-storage is plugged in connector on the router, it is possible to check if the hard disk was recognized with the following command:

```
dmesg | grep USB
```

Something similar to this should show up:

```
hub.c: new USB device 01:03.2-1, assigned address 2
scsi0 : SCSI emulation for USB Mass Storage devices
  Vendor: WDC WD10  Model: EAVS-00D7B1       Rev:
  Type:   Direct-Access                      ANSI SCSI revision: 02
```

[2]Some might wonder why ext2 is used instead of ext3. This is due to the fact that ext3 is a journaling filesystem. Since the partitions are encrypted ext3 would detect that something is wrong and would destroy the filesystem while trying to repair it.

```
Attached scsi disk sda at scsi0, channel 0, id 0, lun 0
SCSI device sda: 1953525168 512-byte hdwr sectors (1000205 MB)
Partition check:
 /dev/scsi/host0/bus0/target0/lun0: p1 p2 p3 p4
WARNING: USB Mass Storage data integrity not assured
USB Mass Storage device found at 2
```

First of all we need to know the device name of the USB-stick on the system:

```
# This command lists all partitions of all harddisks:
fdisk -l
```

I partitioned and formated the device on my linux desktop (much faster than the router):

```
root@helber:~# fdisk -l

Disk /dev/sda: 160.0 GB, 160041885696 bytes
255 heads, 63 sectors/track, 19457 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x9f71e00f

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *           1       19272   154802308+  83  Linux
/dev/sda2           19273       19457     1486012+   5  Extended
/dev/sda5           19273       19457     1485981   82  Linux swap / Solaris

Disk /dev/sdb: 1000.2 GB, 1000204886016 bytes
255 heads, 63 sectors/track, 121601 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0x539fda0b

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1      121601   976760001    c  W95 FAT32 (LBA)
root@helber:~#
```

On my setup the devicename of my 1TB USB-storage was /dev/sdb. On the router the devicename is usually something similar to /dev/scsi/host0/bus0/target0/lun0/disc.

To be able to partition the device all mounted partitions of this device need to be unmounted. This is not necessary on a router, since there is usually no automount. But if you do the partitioning on a linux desktop, your system might mount the device automatically.

```
umount /dev/sdb1
```

Let's partition the USB-storage

```
# On my linux desktop:
fdisk /dev/sdb

# On the router it would probably be something similar to this:
fdisk /dev/scsi/host0/bus0/target0/lun0/disc

# Partition; delete
p d

# new; primary partition; Partition-Nr: 1;
# First cylinder: 1; Last cylinder: 12

n p 1 1 12
# new; primary partition; Partition-Nr: 2;
# First cylinder: 13; Last cylinder: 37
n p 2 13 37

# Set partition to Swap partition
t 2 L 82

# new; primary partition; Partition-Nr: 3;
# First cylinder: 26; Last cylinder: 60813
n p 3 38 60819

# new; primary partition; Partition-Nr: 4;
# First cylinder: 60814; Last cylinder: 121601
n p 60820 121601

# print the partition table
p
# Write changes to disk!
w
```

The p-command in fdisk should print something similar:

```
Disk /dev/sdb: 1000.2 GB, 1000204886016 bytes
255 heads, 63 sectors/track, 121601 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
Disk identifier: 0xb73a3211

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1          12       96358+  83  Linux
/dev/sdb2              13          37      200812+  82  Linux swap / Solaris
/dev/sdb3              38       60819   488231415   83  Linux
/dev/sdb4           60820      121601   488231415   83  Linux
```

After the USB-storage was partitioned, those partitions need to be formated with the ext3 filesystem. This can be done with the following commands:

```
# Now the USB-stick is beeing formated:
mkfs.ext3 /dev/sdb1
```

On the output of the console something similar to this should pop up:

```
mke2fs 1.40.2 (12-Jul-2007)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
244800 inodes, 489468 blocks
24473 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=503316480
15 block groups
32768 blocks per group, 32768 fragments per group
16320 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912

Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 20 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

I recommend ext3 on the router USB-stick also for users, that are using windows for their desktop-pc, cause ext3 has the advantag it is a journaling filesystem and therefore the data is much more secure than with using FAT32. Independent of the filesystem Linux, Windows and Mac can access samba-shares on the router.

8.4 Mount USB-storage

Following the USB-storage can be mounted:

```
mkdir /mnt/part1
mount -t ext3 /dev/scsi/host0/bus0/target0/lun0/part1 /mnt/part1
cd /mnt/part1
ls -la
```

For further information please refer to: [USB storage howto] [FStab]

```
echo "config global automount
        option from_fstab 1
        option anon_mount 1

config global autoswap
        option from_fstab 1
        option anon_swap 0

#config mount
#       option target   /home
#       option device   /dev/sda1
#       option fstype   ext3
#       option options  rw,sync
#       option enabled  0
#       option enabled_fsck 0

config swap
        option device   /dev/scsi/host0/bus0/target0/lun0/part2
        option enabled  0

config mount
        option target   /mnt/part1
        option device   /dev/scsi/host0/bus0/target0/lun0/part1
        option fstype   ext3
        option options  rw,sync
        option enabled  1

config mount
        option target /mnt/part2-crypted
        option device /dev/scsi/host0/bus0/target0/lun0/part3
        option fstype ext2
        option options defaults,loop=/dev/loop/0,encryption=AES256
        option enabled 0

config mount
        option target /mnt/part3-crypted
        option device /dev/scsi/host0/bus0/target0/lun0/part4
        option fstype ext2
        option options defaults,loop=/dev/loop/1,encryption=AES256
        option enabled 0" > /etc/config/fstab

opkg install block-mount
rm /etc/fstab
ln -s /tmp/fstab /etc/fstab
/etc/init.d/fstab start
```

8.5 Activate swap partition

Precondition: Section 8.3 on page 21.

```
mkswap /dev/scsi/host0/bus0/target0/lun0/part2
swapon /dev/scsi/host0/bus0/target0/lun0/part2
free
```

9 Packete installieren

9.1 Installation von Packeten auf USB-storage vorbereiten

Precondition: Section 8.4 on page 24.
Mit folgenden Befehlen wird auf dem USB-Stick ein Bereich eingerichtet auf dem Router-Dienste installiert werden können. http://wiki.openwrt.org/PackagesOnExternalMediaHowTo

```
# Ordner anlegen:
mkdir /mnt/part1/jffs_export
# USB-Stick in OPKG-Konfiguration eintragen:
echo "dest usb /mnt/part1/jffs_export" >> /etc/opkg.conf

# USB-Stick beim Hochstarten automatisch mounten:
echo 'mount /dev/scsi/host0/bus0/target0/lun0/part1 /mnt/part1' >> /etc/init.d/extern
echo 'swapon /dev/scsi/host0/bus0/target0/lun0/part2' >> /etc/init.d/externalmount
chmod +x /etc/init.d/externalmount
ln -s /etc/init.d/externalmount /etc/rc.d/S98externalmount

echo "
config mount
        option target   /mnt/part1
        option device   /dev/scsi/host0/bus0/target0/lun0/part1
        option fstype   ext3
        option options  rw,sync
        option enabled  1" >> /etc/config/fstab

mkdir /mnt/part1/bin
cd /mnt/part1/bin
wget http://www.helber.it/fileadmin/Projekte/Router/opkg-link
chmod +x opkg-link
mv opkg-link opkg-link.sh
ln -s /mnt/part1/bin/opkg-link.sh /usr/bin/opkg-link.sh
```

Da der Router nur begrenzt Flashspeicher zur verfügung stellt, ist dies eine gute Möglichkeit um den Router um jede Menge Funktionalität erweitern zu können.

9.2 Loop AES

Precondition: Section 8.3 on page 21.
Let's install all the necessary packages. [Filesystem Encryption] Usually those packages are installed on the root of the filesystem, since we can't install packages for USB-access on an USB storage device ... *g*

```
opkg update
# Should be installed on USB-fs:
opkg install -dest usb kmod-crypto-core
opkg install -dest usb kmod-crypto-aes
opkg install -dest usb kmod-loop-aes
opkg-link.sh add kmod-crypto-core
opkg-link.sh add kmod-crypto-aes
opkg-link.sh add kmod-loop-aes
opkg install -dest usb libext2fs
opkg install -dest usb e2fsprogs
opkg install -dest usb aeslosetup
opkg-link.sh add aeslosetup
opkg-link.sh add e2fsprogs
opkg-link.sh add libblkid
opkg-link.sh add libext2fs
opkg-link.sh add libuuid
opkg install -dest usb aesswap-utils
opkg-link.sh add aesswap-utils
#For the encrypted partitions necessary
opkg install kmod-fs-ext2

# Should be installed on root:
opkg install -force-overwrite aesmount-utils
reboot
```

Some good links:
http://archiv.raid-rush.ws/t-19217.html
http://www.pro-linux.de/t_system/loop-aes.html ToDo JHR

```
mkfs.ext3 /dev/scsi/host0/bus0/target0/lun0/part1
aeslosetup -e AES256 /dev/loop/0 /dev/scsi/host0/bus0/target0/lun0/part3
mkfs.ext2 /dev/loop/0
aeslosetup -d /dev/loop/0
aeslosetup -e AES256 /dev/loop/1 /dev/scsi/host0/bus0/target0/lun0/part4
mkfs.ext2 /dev/loop/1
aeslosetup -d /dev/loop/1

#aeslosetup -e AES256 /dev/loop/0 /dev/scsi/host0/bus0/target0/lun0/part3
#mount -t ext2 /dev/loop/0 /mnt/part2-crypted/
```

```
#aeslosetup -e AES256 /dev/loop/1 /dev/scsi/host0/bus0/target0/lun0/part4
#mount -t ext2 /dev/loop/1 /mnt/part3-crypted/
```

Known Error:

```
Error:
/dev/loop/0: No such file or directory

In /etc/init.d/boot there is a line similar to this:
load_modules /etc/modules.d/*

That is called before the /mnt/part1 is mounted.

Solution 1:
insmod /mnt/part1/jffs_export/lib/modules/2.4.35.4/loop.o

Solution 2:
Put this command into a later called shell-script.
E.G.:
echo 'mount /dev/scsi/host0/bus0/target0/lun0/part1 /mnt/part1' >> /etc/init.d/extern
echo 'insmod /mnt/part1/jffs_export/lib/modules/2.4.35.4/loop.o' >> /etc/init.d/exter
chmod +x /etc/init.d/externalmount
# ToDo JHR

echo "
config mount
        option target   /mnt/part2-crypted
        option device   /dev/scsi/host0/bus0/target0/lun0/part3
        option fstype   ext2
        option options  defaults,loop=/dev/loop/0,encryption=AES256
        option enabled  0

config mount
        option target   /mnt/part3-crypted
        option device   /dev/scsi/host0/bus0/target0/lun0/part4
        option fstype   ext2
        option options  defaults,loop=/dev/loop/1,encryption=AES256
        option enabled  0" >> /etc/config/fstab

mount /mnt/part2-crypted
#(Passworteingabe)
mount /mnt/part3-crypted
#(Passworteingabe)

umount /mnt/part2-crypted;
```

```
umount /mnt/part3-crypted;
aeslosetup -d /dev/loop/0
aeslosetup -d /dev/loop/1
```

9.3 Samba

First of all the package list needs to be updated for being able to install samba:

```
opkg update

opkg -dest usb install samba-server samba-common
opkg-link.sh add samba-server
opkg-link.sh add samba-common

opkg list_installed

mkdir /mnt/part1/etc
mkdir /mnt/part1/log
```

In the file /etc/samba/smb.conf the folder /tmp is shared by default. Therefore the changes are lost after each reset of the router. To prevent this smb.conf should be modified before starting samba:

```
ls -la /etc/samba
# smb.conf  -> /mnt/part1/jffs_export/etc/samba/smb.conf
# smbpasswd -> /mnt/part1/jffs_export/etc/samba/smbpasswd

echo "[global]
 syslog = 0
 syslog only = yes
 workgroup = OpenWrt
 server string = OpenWrt Samba Server
 security = share
 browseable = yes
 guest only = no
 log level = 1
 log file = /mnt/part1/log/samba.log
 max log size = 100
 smb passwd file=/etc/samba/smbpasswd
 encrypt passwords = yes
 guest account = nobody
 local master = yes
 name resolve order = lmhosts hosts bcast
 dns proxy = no
 unix extensions = no
 client lanman auth = yes
```

```
 client plaintext auth = yes

[public]
 comment = accessible (but not writeable) to everybody
 path = /mnt/part2-crypted/Smb-Public
 browseable = yes
 public = yes
 writeable = no

[john]
 comment = private
 path = /mnt/part2-crypted/Smb-John
 browseable = yes
 public = no
 writeable = yes
 valid users = john

[upload]
 comment = scratch: accessible and writeable to everybody
 path = /mnt/part2-crypted/Smb-Upload
 browseable = yes
 public = yes
 writeable = yes
 guest ok = yes" > /mnt/part1/jffs_export/etc/samba/smb.conf

ln -s /mnt/part1/jffs_export/etc/samba/smb.conf /etc/samba/smb.conf

# This will add a user john with password john as system user
echo "john:$1$cenDCinJ$cm2lKiOUOA7A9A4wkIs3B0:1000:1000:nobody:/var:/bin/false" >> /e

# Change the owner of the folder to the user recently created
chown john /mnt/part2-crypted/Smb-John

# Change the password of the system user
passwd john

# Now we add the user john as a samba user as well
touch /mnt/part1/jffs_export/etc/samba/smbpasswd
smbpasswd --help
smbpasswd -a john
```

In /etc/hosts replace:

```
127.0.0.1 localhost.
```

By:

```
127.0.0.1 localhost. OpenWrt
```

Start Samba:

```
/etc/init.d/samba start
```

9.4 Cron

Precondition: Section 9.4.4 on page 34.
http://www.macsat.com/macsat/content/view/20/30/
The cron deamon (crond) is a standard part of OpenWRTs version of BusyBox. This means that there is nothing to install, just a few things to setup.

Since crontab can be a pain to use, lets create a few dirs, where we can place scripts that can be ran at specified times:

```
mkdir /etc/cron.5mins
mkdir /etc/cron.hourly
mkdir /etc/cron.daily
mkdir /etc/cron.weekly
mkdir /etc/cron.monthly
```

Now you need to create the file: /etc/crontabs/root like this:

```
echo "# Syntax for lines is : minute hour day month dayofweek command #
*/5  * * * *  /usr/bin/run-parts /etc/cron.5mins
15   * * * *  /usr/bin/transfer.sh
01   * * * *  /usr/bin/run-parts /etc/cron.hourly
02   4 * * *  /usr/bin/run-parts /etc/cron.daily
22   4 * * 0  /usr/bin/run-parts /etc/cron.weekly
42   4 1 * *  /usr/bin/run-parts /etc/cron.monthly" > /etc/crontabs/root
```

As also noted in the file, the syntax for each line is :
minute hour day month dayofweek command

To take an example, the command: /usr/bin/run-parts /etc/cron.weekly is executed at time :
minute : 22
hour : 4
day : *
month : *
dayofweek : *

Where "*" is a wild-card meaning "ALL".
Thus the command is running : Every day of week, Every Month, Every Day at 4:22

Note that cron uses a 24h clock.

Having created this config-file, all that is needed is the command "run-parts" which is not a part of the firmware.
For this purpos I created a small script which I named run-parts and placed in /usr/bin
A script like this will do the job:

```
cd /mnt/part1/bin/
wget http://www.helber.it/fileadmin/Projekte/Router/run-parts
# Remember to chmod the script to be executable :
chmod +x /mnt/part1/bin/run-parts
ln -s /mnt/part1/bin/run-parts /usr/bin/run-parts
ls -la /usr/bin/run-parts
```

All that is left to do, is to ensure the cron system get stated at boot-time. Create the file /etc/init.d/S61crond:

```
echo "#!/bin/sh /etc/rc.common
# Copyright (C) 2006 OpenWrt.org
START=50

start () {
        [ -z \$(ls /etc/crontabs/) ] && exit 1
        mkdir -p /var/spool/cron
        [ -L /var/spool/cron/crontabs ] || ln -s /etc/crontabs /var/spool/cron/cronta
        crond -c /etc/crontabs
}

stop() {
        killall -9 crond
}" > /etc/init.d/cron

# Make the file executable:
chmod +x /etc/init.d/cron
ln -s /etc/init.d/cron /etc/rc.d/S50cron
ls -la /etc/rc.d/S50cron
```

In order to test the cron-system, you could try to make a small script in /etc/cron.5mins called test.sh, like this:

```
echo "#!/bin/sh
date >> /tmp/crontest.txt" > /etc/cron.5mins/test.sh
# Remember to make it excutable:
chmod 755 /etc/cron.5mins/test.sh
```

Try to reboot your router (or call "crond -c /etc/crontabs -b") and let it run for some 10 or 15 minuts. Then you should see a list of timestamps in /tmp/crontest.txt showing when the cron has been running the script:

```
cat /tmp/crontest.txt
```

9.4.1 DynDns

Precondition: Section 9.4 on page 31.

```
cd /etc/cron.5mins/
wget http://www.helber.it/fileadmin/Projekte/Router/dyndns.sh.txt
mv dyndns.sh.txt dyndns.sh
chmod +x dyndns.sh
```

9.4.2 Rsync

```
opkg update
opkg -dest usb install rsync
opkg-link.sh add rsync
opkg-link.sh add libpopt
# Without a swap-partition only very few files can be transfered
opkg install swaputils

opkg install -dest usb dropbearconvert
opkg-link.sh add dropbearconvert
dropbearconvert openssh dropbear /mnt/part3-crypted/siemens-key \
  /mnt/part3-crypted/siemens-key-dropbear

mkdir /mnt/part1/tmp
```

That's how my backup-script looks like:

```
cd /mnt/part1/bin/
wget http://www.helber.it/fileadmin/Projekte/Router/transfer.sh
chmod +x transfer.sh
ln -s /mnt/part1/bin/transfer.sh /usr/bin/transfer.sh
ls -la /usr/bin/transfer.sh
```

To let the cron-daemon call the synchronisation daily:

```
ln -s /usr/bin/transfer.sh /etc/cron.daily/transfer.sh
```

Since rsync first exchanges the filelist from one system to the other, usually ram and swap are used heavily if partitions with lots of files are being synced. To get updated information regarding the free memory you can use the following commands. Can consider that this can only be terminated from a different console with "killall sh":

```
echo "while true
do
clear
free
sleep 5
done" > /mnt/part1/bin/freeloop.sh
chmod +x /mnt/part1/bin/freeloop.sh
ln -s /mnt/part1/bin/freeloop.sh /usr/bin/freeloop.sh

# Can only be terminated from a different console with ‘‘killall sh’’
freeloop.sh
```

9.4.3 Zeitzone

Mit folgendem Befehl stellen wir die Zeitzone auf Berlin/Germany um:

```
echo "CET-1CEST-2,M3.5.0/02:00:00,M10.5.0/03:00:00" > /etc/TZ
# Howto enter the correct date and time?
date --help
# e.g.: date '2009-03-04 23:15'
# e.g.: date 2010.07.17-14:55
```

9.4.4 RDate

```
echo "#!/bin/sh
/usr/sbin/rdate -s -p ptbtime1.ptb.de
/usr/sbin/rdate -s time-a.nist.gov
echo \"CET-1CEST-2,M3.5.0/02:00:00,M10.5.0/03:00:00\" > /etc/TZ" > /etc/init.d/rdate

# DonÄT forget to make it executable
chmod +x /etc/init.d/rdate
ln -s /etc/init.d/rdate /etc/rc.d/S60rdate
ls -la /etc/init.d/rdate
```

Now running /etc/rc.d/S60rdate or rebooting your router, will set the time correctly according to your current timezone.
Now we are ready to configure cron.

9.4.5 OpenVPN

Precondition: Section 9.4.4 on page 34.
[OpenWrt OpenVPN] [DDWrt OpenVPN] First of all OpenVPN needs to be installed:

```
opkg -dest usb install openvpn
opkg-link.sh add openvpn
```

```
opkg-link.sh add kmod-tun
opkg-link.sh add libopenssl
opkg-link.sh add zlib
opkg-link.sh add liblzo

mkdir /mnt/part1/OpenVPN
cd /mnt/part1/OpenVPN
openvpn --genkey --secret static.key
ln -s /mnt/part1/OpenVPN/static.key /etc/static.key

echo "
config rule
       option src              wan
       option dest_port        1194
       option target           ACCEPT
       option proto            udp

config rule
        option src              wan
        option dest_port        1194
        option target           ACCEPT
        option proto            tcp" >> /etc/config/firewall
uci commit firewall
/etc/init.d/firewall restart

mkdir /etc/openvpn
cd /etc/openvpn
wget http://www.helber.it/fileadmin/Projekte/Router/openvpn-startupscript
mv openvpn-startupscript startupscript
cd /mnt/part1/OpenVPN
wget http://www.helber.it/fileadmin/Projekte/Router/server.ovpn

ln -s /mnt/part1/OpenVPN/server.ovpn /etc/openvpn/server.ovpn
ln -s /mnt/part1/OpenVPN/static.key /etc/openvpn/secret.key
iptables -A INPUT   -i tap+ -j ACCEPT
iptables -A FORWARD -i tap+ -j ACCEPT
chmod +x /mnt/part1/OpenVPN/startupscript

echo "config interface vpn
        option ifname   tap0
        option proto    static
        option netmask  255.255.255.0
        option dns      ''
        option gateway  ''
        option ipaddr   10.10.10.1" >> /etc/config/network
```

```
echo "
config 'zone'
       option 'name'           'vpn'
       option 'input'          'ACCEPT'
       option 'output'         'ACCEPT'
       option 'forward'        'ACCEPT'

config 'forwarding'
       option 'src'            'lan'
       option 'dest'           'vpn'" >> /etc/config/firewall

echo "#!/bin/sh /etc/rc.common

START=65
STOP=35

start() {
                /etc/openvpn/startupscript up
                openvpn --daemon --config /etc/openvpn/server.ovpn
}

restart() {
                $0 stop
                sleep 3
                $0 start
}

reload() {
                killall -SIGHUP openvpn
}

stop() {
                killall openvpn
                sleep 3
                /etc/openvpn/startupscript down
}" > /etc/init.d/openvpn2
chmod +x /etc/init.d/openvpn2


vi /etc/config/firewall
/etc/init.d/firewall restart
uci show network; uci show firewall
/mnt/part1/OpenVPN/startupscript up
ifconfig tap0 10.10.10.1 netmask 255.255.255.0
route add -net 192.168.4.0 netmask 255.255.255.0 gw 10.10.10.2 tap0
```

```
# LAN Steffen: 192.168.4.0/24
openvpn --daemon --config /etc/openvpn/server.ovpn

#brctl show
#brctl addif br-lan tap0
#ifconfig tap0 0.0.0.0 promisc up
```

Known problems:
First decrypt, then encrypted transfer, than encrypt
Better transfer stream of filesystem ... but different clusters might be corrupt on the different hdd

9.4.6 Socks5

opkg -dest usb install srelay opkg-link.sh add srelay echo "# allow local subnet to access socks proxy 192.168.1.0/24 any - 127.0.0.1/32 any -" > /etc/srelay.conf srelay -h srelay -c /etc/srelay.conf -r -s

9.4.7 WOL

```
opkg -dest usb install etherwake
opkg-link.sh add etherwake

echo "etherwake 00:30:05:8a:51:ff" > /bin/wol.sh
chmod +x /bin/wol.sh
/bin/wol.sh
ping 192.168.4.240
```

9.4.8 QoS

Um QoS[3] nutzen zu können, m"ussen wir TC installieren und ein paar Daten auf den Router kopieren:

```
ipkg update

# Da die Installation von TC scheinbar extrem viel
# System-Resourcen benötigt, schaufeln wir ein paar frei
killall httpd
killall crond
killall udhcpc

ipkg install kmod-sched kmod-ipt-conntrack iptables-mod-conntrack kmod-ipt-ipopt \
iptables-mod-ipopt kmod-ipt-extra iptables-mod-extra iptables-extra tc iptables-mod-filter

#ipkg install http://www.helber.it/fileadmin/Projekte/Router/qos-scripts_0.9.1-1_mipsel.ipk
#ipkg install http://www.helber.it/fileadmin/Projekte/Router/tc_2.6.11-050330-1_mipsel.ipk
```

[3] Quality of Service

```
cd /etc/l7-protocols/
wget http://www.helber.it/fileadmin/Projekte/Router/pattern.tar
tar -xvf pattern.tar
rm pattern.tar

cd /usr/bin
# Danach richten wir uns einige nette Shell-Skripte ein,
# welche uns einige coole Features einrichten:

# http://www.ipcop-forum.de/forum/viewtopic.php?t=4749&postdays=0&postorder=asc&start=120
# http://voip-info.org/tiki-index.php?page=QoS%20Linux%20with%20HFSC
# http://www.ip-phone-forum.de/archive/index.php/t-87301.html
wget http://www.helber.it/fileadmin/Projekte/Router/qos.sh

#wget http://www.helber.it/fileadmin/Projekte/Router/qos2.sh

# SipShaper: QoS mit HFSC - bei mir klappt's super! (HFSC)
# http://www.ip-phone-forum.de/showthread.php?t=72590
wget http://www.helber.it/fileadmin/Projekte/Router/qos3.sh

wget http://www.helber.it/fileadmin/Projekte/Router/qos4.sh
wget http://www.helber.it/fileadmin/Projekte/Router/qos5.sh

# HTB
# http://www.ip-phone-forum.de/showthread.php?t=95566
wget http://www.helber.it/fileadmin/Projekte/Router/load_shaper_modules
wget http://www.helber.it/fileadmin/Projekte/Router/qos6.sh

# NSLU2
# http://www.nslu2-linux.org/wiki/HowTo/EnableTrafficShaping
# Verweis von: http://www.ip-phone-forum.de/showthread.php?p=344900#post344900
wget http://www.helber.it/fileadmin/Projekte/Router/qos7.sh

chmod +x /usr/bin/*.sh

# ordentliche tc-Version die auch HTB kann
# im Repository war beim schreiben der Doku V2.0 aktuell
#wget http://www.helber.it/fileadmin/Projekte/Router/htb3.6-020525.tgz
#gzip -d htb3.6-020525.tgz
#tar -xvf htb3.6-020525.tar
```

Um den Wert MYUMAX in der Datei "/usr/bin/qos.sh" eintragen zu können, muss dieser erst ermittelt werden. Dazu gibt man auf der Konsole folgende Kommandos ein:

```
i=1400
o=0
while [ $o -eq 0 ]
do
  echo $i
  ping -s $i -c 1 ftp.t-online.de > /tmp/ping
  o=‘cat /tmp/ping | grep -c '0 packets received'‘
  i=‘expr $i + 1‘
done
i=‘expr $i + 26‘
echo Es muss $i eingetragen werden!
rm /tmp/ping
```

Der Wert UPRATEJAN in der Datei "/usr/bin/qos.sh" muss auf den maximalen Upload (kBit) eingestellt werden.

Um zu überprüfen welche Pakete wie markiert werden kann man folgenden Befehl verwenden:

```
#wget http://www.tom-e.de/download/iftop016.tar.gz
```

```
#tar xfz iftop016.tar.gz
#cd iftop016
#./install -i
#iftop -i ppp0

grep 192.168.1.100 /proc/net/ip_conntrack
```

9.5 Shell-Skripte

Wenn man folgende Befehle der Reihe nach kopiert und in der Konsole einfügt, erhält man diese Funktionalität dazu:

- whatismyip.sh: Gibt die IP-Adresse des Routers aus
- switchmode.sh {switch | router | routerwan | lanparty}
  Ich nutze im Wohnheim den Modus routerwan. Einstellungen am Modus kann man in der Datei "/etc/init.d/S02mode_switch" vornehmen. Man sollte natürlich wissen was man tut ;)
- deutlich einfacheres Verwalten der Firewallregeln!
  Es ist zwar möglich Änderungen an der Datei "/usr/bin/firewall.user.sh" vorzunehmen. Allerdings wird diese Datei bei jedem Start des Routers durch das "/etc/init.d/S49startup"-Skript aufgerufen. Sobald mal ein Fehler in der "firewall.user.sh" gemacht wurde bringt auch ein Reset des Routers nichts mehr. Daher empfehle ich Änderungen zunächst in der Datei "firewall.user.test.sh" zu machen und diese dann manuell auszuführen (TESTEN!). Den Grundegedanken dazu habe ich von [Simple Firewall].
  Ich habe allerdings verschiedenes angepasst und verbessert.
  Die Firewall-Konfiguration lässt sich mit folgenden Befehlen anzeigen:

  ```
  iptables -t nat -L
  iptables -L
  ```

- Cron-Jobs werden möglich gemacht. Ein Cron-Job wird eingerichtet, der stündlich die Uhrzeit übers Netz aktualisiert.
- Aliase werden angelegt, die das Arbeiten mit dem System vereinfachen sollen.
- Feste Hosteinträge für den DNS werden gesetzt.
- cooles Skirpt um Dateien auf Linux-Rechner zu übertragen "tarsend.sh"
- cooles Skirpt um Dateien von Linux-Rechner zu übertragen "tarrecv.sh"
- cooles Skript um die Signalstärke zu einem verbundenen Client zu testen "signal-staerke.sh"

Diese Befehle einfach kopiert und in der Konsole einfügt.

```
cd /usr/bin
wget http://www.helber.it/fileadmin/Projekte/Router/signalstaerke.sh
wget http://www.helber.it/fileadmin/Projekte/Router/tarsend.sh
wget http://www.helber.it/fileadmin/Projekte/Router/tarrecv.sh
wget http://www.helber.it/fileadmin/Projekte/Router/whatismyip.sh
wget http://www.helber.it/fileadmin/Projekte/Router/switchmode.sh
wget http://www.helber.it/fileadmin/Projekte/Router/fwlib.sh
wget http://www.helber.it/fileadmin/Projekte/Router/firewall.user.sh
wget http://www.helber.it/fileadmin/Projekte/Router/firewall.user.test.sh
cd /etc/init.d
wget http://www.helber.it/fileadmin/Projekte/Router/S49startup
wget http://www.helber.it/fileadmin/Projekte/Router/S02mode_switch
cd /etc
wget http://www.helber.it/fileadmin/Projekte/Router/functions.sh
rm banner
#Und wenn man noch eine schöne Begrüßung nach dem Einloggen haben möchte:
wget http://www.helber.it/fileadmin/Projekte/Router/banner

touch /usr/sbin/cron.minutely
touch /usr/sbin/cron.hourly
touch /usr/sbin/cron.daily
chmod +x /usr/sbin/cron.*
rm /etc/init.d/S60cron
echo "#!/bin/sh" > /etc/init.d/S60cron
echo "/usr/sbin/crond" >> /etc/init.d/S60cron
chmod +x /etc/init.d/S60cron

#echo "#!/bin/sh" > /usr/bin/whatismygw.sh
#echo "netstat -rn | awk '/^0\.0\.0\.0/ {print \$2}'" >> /usr/bin/whatismygw.sh
#chmod +x /usr/bin/whatismygw.sh
#TODO netstat

#echo "#!/bin/sh" > /usr/bin/whatismygwmac.sh
#echo "arping -fq -I vlan1 \`whatismygw.sh\`" >> /usr/bin/whatismygwmac.sh
#echo "cat /proc/net/arp | grep \`whatismygw.sh\` | awk '{print \$4}'" \
# >> /usr/bin/whatismygwmac.sh
#chmod +x /usr/bin/whatismygwmac.sh

echo "[ -f /usr/bin/firewall.user.sh ] && . /usr/bin/firewall.user.sh" >> /etc/init.d/S45firewall
echo "iptables -I INPUT 8 -p tcp --dport 22 -j ACCEPT" >> /etc/init.d/S45firewall

#neu angelegte Skripte ausführbar machen
chmod +x /usr/bin/*.sh
chmod +x /etc/init.d/S49startup
chmod +x /etc/init.d/S02mode_switch
./S49startup
```

Hier gibts eine kleine Zusammenfassung über die verschiedenen Modi:

- Switch mode:
  - WAN deaktiviert, Alle 5 Ports werden fürs LAN genutzt
  - LAN wird per DHCP konfiguriert
  - Das WLAN-Interface wird zum LAN gebridged
    WPA2 Verschlüsselung aktiviert
  - Firewall deaktiviert
  - DHCP server deaktiviert
  - OpenWRT web interface deaktiviert
- Router-Modus:
  - WAN aktiviert (PPPoE)
  - LAN aktiviert: nutzt statische IPs (beim Client einstellen)
  - WLAN-Konfiguration wie im Switch-Modus
  - Alle Dienste aktiviert
- RouterWAN-Modus:
  - WAN aktiviert (DHCP)
  - LAN-Konfiguration wie beim Router-Modus
  - WLAN deaktiviert
  - Firewall aktiviert
  - DHCP server aktiviert
  - OpenWRT web interface aktiviert
- Lanparty-Modus:
  - WAN disabled
  - LAN-Konfiguration wie im Router-Modus
  - WLAN aktiviert: anders konfiguriert als in den vorherigen Modi
  - Firewall deaktiviert
  - DHCP server aktiviert
  - OpenWRT web interface deaktiviert
  - spezieller LANparty web server (httpd2): z.B. zum Pizza bestellen ;)

9.6 MAC address cloning

Bei mir im Wohnheim ist für meinen Anschluss nun dummerweise die Mac-Adresse meines Notebooks eingetragen. Ausschließlich die Mac-Adresse "00:03:0D:02:1D:7A" hat Zugriff auf das Internet. Da es aber schon etwas spät war den Admin raus zu klingeln und ich unabhängig davon auch gar keine Lust hatte ihm zu erzählen, dass ich ab sofort einen Router am Netz habe, hab ich mich über "MAC address cloning in OpenWrt" schlau gemacht. Das funktioniert folgendermaßen:
Um die Mac-Adresse des Routers zu ändern müssen die Variablen "wan_hwaddr" und "et0macaddr" im NVRAM überschrieben werden. Das funktioniert mit folgenden Kommandos:

```
#OpenWrt Kamikaze
ifconfig eth0 ether 00:11:22:33:44:55
# ToDo JHR

# OpenWrt White russian
nvram set wan_hwaddr="00:03:0D:02:1D:7A"
nvram set et0macaddr="00:03:0D:02:1D:7A"
nvram show | grep mac
nvram commit

ifdown wan
ifup wan
```

Da mein LapTop (Windows XP) natürlich nicht die selbe MAC-Adresse haben kann wie der Router (sonst funktioniert gar nichts mehr) musste ich diese auch noch ändern unter "Start > Systemsteuerung > System > Hardware > Geräte-Manager > Netzwerkadapter > [Kartenname] > Eigenschaften > Erweitert > Network Address". Dort stellte ich einfach "00030D021D7B" ein.
Oder unter Linux:

```
ifconfig eth0 ether 00:03:0D:02:1D:7B
```

Und wenn wir schon dabei sind, tragen wir im Router f"ur die entsprechende MAC-Adresse eine feste IP ein:

9.7 SSH-Public-Key-Authentification

Um nicht jedes mal das Passwort eingeben zu müssen, kann man sich wie bei jedem Linux mit einem Key authentifizieren [4]. Zum erstellen solcher Keys ist unter Windows das Tool "puttygen.exe" gut geeignet.

[4] wobei man den Public Key ohne Risiko veröffentlichen kann

Der generierte Public-Key muss dann unter "/etc/dropbear/authorized_keys" und "/etc/-dropbear/id_rsa.pub" abgelegt werden. Z.B. so:

```
echo "ssh-rsa AAAAB3NzaC1yc2EAAAABJQAAAgBxCRPq4/nc08NFEzWF8y17/jWBorP"\
"WKb/nyZs+bcq/pa5x23tZO00Mb73tzN5APq1h1xr7cUzQLajSMHm3p0Ie6slbamQqwgZ"\
"TL012oSvgdndqjmJM3A7hdi4yHb6A0hiVEa2OIlLWCsJPnqpZmOCsTjurtJH2QNN7Wp3"\
"v2ycaD+gdazIf5wtq8dBRE4EE8pAba88yhr3/Gi/Wqc3N1UGRSMDp04xosm1SFTlDc4d"\
"wV3zWflPFoplSErf+8L18xCfTS6hpz42bwRcbRd3wbEgvRoV4vOZVCjL+R9nlL79Kv9b"\
"2WuR59IK72ke58DlQ9m1HxVr53xSoK+ToFsazoVLwt9AeCQbtKJNUZrI1c7Ob6GAERm1"\
"ANGr4qqsddECu219u12sbpkXzrP1j+GXVLq6mzR3vIP44MwEoSaBPik/NCrekjbdNgvE"\
"CsvZ86KyR6Hf8KvyALyTSBuo/sAJIPeOHlAZ/8DDr5/uDtO7EJL82prI0LnrLyMKUqhs"\
"EZ68LUFouM/c4RMsT9QHmYtNMYRtked1MOKHUXsEWgm65Re1ozMDNIKm7jgwMyXTbrXC"\
"CZCeZnd8ca07fpCyT5z6ivFh9jqvdRa7rDK2QTpISXRDFZl1jvdYkmIbnd6BaB4OrWmf"\
"NuNO/WY7iMcNAJqRK91Ml2UyYohcRd1ljaWur"\
"5YGrwQ== rsa-key-20081228" >> /etc/dropbear/authorized_keys
chmod 0600 /etc/dropbear/authorized_keys
```

10 Backup

After all this work it is time to save a backup of the system. Therefore execute the following command on a linux workstation which is connected with the router via ethernet:

```
IP=192.168.1.1
ssh root@$IP -C 'mount -o remount,ro /dev/mtdblock/4 \
/jffs ; dd if=/dev/mtdblock/1 ; mount -o remount,rw \
/dev/mtdblock/4 /jffs' > 2009-10-21-router-backup.trx
```

After a system-crash the router can be restored easily:

```
IP=192.168.1.1
sudo atftp --tftp-timeout 1 --trace -p -l image.trx $IP
```

11 Viel Spaß!

Ich wünsche viel Spaß mit dem Router!
Wenn ein Leser dieser Dokumentation erfolgreich eine WebCam, ein Bluetooth-Gerät oder etwas ähnliches an dem Router anschließt, wäre ich über eine kurze Info (wenn möglich kurze Anleitung) dankbar. Ich würde diese Infos dann in dieser Anleitung aufnehmen.
In meinem Wohnheim sollte der Router über den WAN-Port auch Dateifreigaben weiter leiten. Wenn ich mal Zeit und Lust habe, werde ich das Problem lösen und hier dokumentieren.

12 FAQ

Was tun wenn man richtig Blödsinn gebaut hat und nichts mehr funktioniert?!
Dann entfernt man die Spannungsversorgung des Routers, wartet kurz und steckt sie wieder ein, während man die RESET-Taste gedrückt hält. Daraufhin flimmert die WLAN-LED am Router kurze Zeit sehr schnell. Dabei werden alle Einstellungen auf den Ausgangszustand zurück gesetzt (auch boot_wait *g*). Sobald der Router neu startet (dass erkennt man daran, das Die LEDs von WAN & LAN1-4 kurz aufleuchten), spielt man per TFTP ein frisches Image ein. Danach sollte wieder ein Zugriff per SSH oder Telnet möglich sein.

Was tun wenn jedesmal nach dem Start von firstboot das System am A**** ist?
Kopier dir die Datei "/rom/bin/firstboot" nach "/"[5] und lösche die 64. Zeile. "test -f etc/dropbear/dropbear_dss_host_key && rm etc/init.d/S*telnet". Dadurch bleibt der passwortfreie Telnetzugang zum System immer erhalten. Wenn das System wieder stabil bootet und man sich per SSH einloggen kann löscht man die Datei "/etc/init.d/S*telnet" einfach von Hand[6].

13 Anhang

14 Quellen

References

[ASUS WL-500gp V2] http://wiki.openwrt.org/oldwiki/openwrtdocs/hardware/asus/wl500gpv2

[Client Mode Wireless] http://wiki.openwrt.org/doc/howto/clientmode

[Bridged Client] http://wiki.openwrt.org/doc/recipes/bridgedclient

[Simple Firewall] http://wiki.openwrt.org/SimpleFirewall

[Port Weiterleitung] http://wiki.opennet-initiative.de/index.php/Portforwarding

[USB storage howto] http://wiki.openwrt.org/UsbStorageHowto

[IP publishing] http://nuwiki.openwrt.org/oldwiki/ddnshowto

[Filesystem Encryption] http://loblog.wordpress.com/2008/05/26/filesystem-encryption-on-openwrt-kamikaze-kernel-24/

[5] cp /rom/bin/firstboot /
[6] rm /etc/init.d/S*telnet

[FStab] http://wiki.openwrt.org/doc/uci/fstab

[OpenWrt OpenVPN] http://wiki.openwrt.org/OpenVPNHowTo

[DDWrt OpenVPN] http://www.dd-wrt.com/wiki/index.php/OpenVPN-Tunnel_Server_/_Client,_Netzwerke_verbinden#Server_Config_.28WRT54_-_LINUX.29

[HowTo OpenWRT] http://wiki.openwrt.org/CategoryHowTo?action=show&redirect=OpenWrtHowTo

15 Marken

Diese Dokumentation beinhaltet eingetragene Marken oder Marken der jeweiligen Eigentümer.
Einige Firmen- und/oder Produktbezeichnungen in dieser Dokumentation sind Warenzeichen und/oder eingetragene Warenzeichen ihrer jeweiligen Besitzer.

16 Haftungsausschluss

Die Vollständigkeit und verlässlichkeit der in dieser Dokumentation enthaltenen Informationen wurden sorgfältig überprüft. Für die Richtigkeit der Angaben kann ich jedoch keine Gewähr übernehmen und für eventuelle Schäden nicht haften. In keinem Fall bin ich Ihnen gegenüber haftbar für Schäden, die auf die Verwendung oder den anderweitigen Einsatz dieser oder anderer Dokumentationen zurückzuführen sind.